

# ‘ALMOST SURE’ CHAOTIC PROPERTIES OF MACHINE LEARNING METHODS

NABARUN MONDAL AND PARTHA P. GHOSH

**ABSTRACT.** It has been demonstrated earlier that universal computation is ‘almost surely’ chaotic. Machine learning is a form of computational fixed point iteration, iterating over the computable function space. We showcase some properties of this iteration, and establish in general that the iteration is ‘almost surely’ of chaotic nature. This theory explains the observation in the counter intuitive properties of deep learning methods. This paper demonstrates that these properties are going to be universal to any learning method.

## 1. MOTIVATION

The motivation of the current paper is two fold.

One of the authors of the current paper was using iterative machine learning to crack cipher code in late 1990s. While doing so he came to the astounding realisation that the resulting learned function is not *convergent* (definition A.3) at all. What is really meant is that : data points  $x_n$  which were accepted up-to iteration  $n$  ( points in the set  $S_n$  ) would fly away eventually after some more iteration at  $m$  ( $S_m$  with  $m > n$ ):

$$\nexists n > N \text{ such that } S_{n+1} \subseteq S_n \text{ and } S_n \subseteq S_{n+1}$$

In other words none of the **lim sup**  $S_n$  and **lim inf**  $S_n$  exists and therefore :

$$\nexists S = \lim_{n \rightarrow \infty} S_n$$

In essence the iteration was showing converging and then sudden diverging behaviour. While he communicated this finding to one of pioneers in the image processing domain - he was simply baffled. In fact the trivia is - he simply remarked : “*I have no idea - what to respond!*”. While this behavior was never fully

---

2010 *Mathematics Subject Classification.* Primary 03D10 ; Secondary 65P20,68Q05,68Q87,68T05.

*Key words and phrases.* Turing Machines ; Universal Computation ; Chaos ; With Probability One ; Aleph Numbers ; Self Similarity ; Fractal ; Machine Learning ; Deep Learning .

Nabarun Mondal : Dedicated to my missing geometry teacher Dr. Sushanta Mondal; my parents : Tapan and Sabita Mondal; Big thanks to : Abhishek Chanda and Shweta Bansal : You all have been constant support.

Partha. P. Ghosh : Dedicated to my parents and family, without their presence we are nothing.

understood by the author then, recent theory by the same author(s) [11] seems to have an explanation to it.

A very recent (yet to be published) paper in arxiv [12] discusses interesting properties in the deep learning. It became immediately apparent to the authors that these two phenomenons are essentially connected.

This paper tries to explain these phenomenons in the light of the discovery of chaos *almost surely happening in computation* (theorem A.2) [11].

## 2. THE NATURE OF LEARNING THEORY

Learning theory is really about classification. As Vapnik [1] pointed it out - it is all about finding a classifier function  $f : X \rightarrow \{0, 1\}$  such as to isolate positive samples ( accept ) from negative ones ( reject ), aided by training set  $T_S$  ; which is a set of ordered pairs  $T_S = \{(x, y)\}$  with  $x \in X$  and  $y \in \{0, 1\}$ . Given  $x \in A$  then  $y = 1$  else  $y = 0$ , with  $A \subset X$  being the accept set.

**2.1. Learned Function as Indicator of Accept Set.** We can say that to learn  $A$  one needs to isolate the set  $A \subset X$  using training data sets  $T_S$ . We need to learn the function  $f_L$  with  $x \in X$  :

$$f_L(x, T_S) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{if } x \in A^C \end{cases}$$

Clearly then, the function we want to learn, is the indicator function [2] of the accept set  $A$ , that is:

$$f_L(, T_S) \equiv 1_A$$

Then we can simply suggest that machine learning is all about finding the positive samples ( accept ) sets indicator function  $1_A$ . This formulation has a problem, because we really do not know that whether or not  $1_A$  is computable. The set of computable functions in this paper would be designated as  $\mathbb{C}$ . If  $1_A \notin \mathbb{C}$  we assume existence of a sequence of computable functions  $\langle C_n \rangle$  which converges [2] to  $1_A$ .

$$\lim_{n \rightarrow \infty} C_n = 1_A$$

This indeed is a very strong axiom - and in the very next section (3) we shall see that this almost surely not holding to be true.

**2.2. Learning theory as Iterative Maps.** We need to generate this sequence of functions  $\langle C_n \rangle$  in a computable way. That immediately gives :

$$C_{n+1} = \mathcal{L}(C_n)$$

the guiding equation of the learning. There has to be a fixed point computable iteration function  $\mathcal{L}$  such that it can take a computable function, returns another one, such that the orbit (definition A.5)  $\langle C_n \rangle$  eventually converges to “ $1_A$ ”.

In effect what we seek is :

$$\lim_{n \rightarrow \infty} C_n = 1_A = \mathcal{L}(1_A)$$

with  $1_A$  is the fixed point of the iteration (definition A.1). All ML and Deep Learning methods can be abstracted in this form, it is in effect a simple tautology.

NOTE however that the function  $\mathcal{L} : \mathbb{Q} \rightarrow \mathbb{Q}$  iteration might not be able to reach  $1_A$  because in general the set of functions  $f \in \mathbb{R}^{\mathbb{R}}$ , which would be discussed in the next section.

**2.3. The Cut Function Approximation of  $1_A$ .** It is a decision that whether or not  $x \in X$  with  $1_A(x) = 1$  when  $x \in A$  and  $1_A(x) = 0$  when  $x \notin A$ . Therefore, clearly  $1_A$  partitions the input space  $X$ . The question of mechanism of computable partitioning requires a definition.

**Definition 2.1. Binary Decision Tree.**

*A binary tree with each nodes having two decider Turing Machine (definition A.11) from a finite set of decider machines  $(D_f, D_d) \in \mathbb{D}$  such that:*

$$D_f : \mathbb{Q} \rightarrow \mathbb{Q} \text{ and } D_d : \mathbb{Q} \rightarrow \{0, 1, l, r\}$$

*Given  $x \in X$  with  $X \subseteq \mathbb{Q}$  is the input to the current node, let  $y = D_f(x)$  and  $d = D_d(y)$ . If  $d = 0$  the input is rejected and the system is halted. If  $d = 1$  the input is accepted and the system is halted. If  $d = l$  then  $y$  is passed as input to the left child ; if no left child exists, reject input  $x$  and halt. If  $d = r$  then  $y$  is passed as input to the right child ; if no right child exists, reject input  $x$  and halt.*

This structure (definition 2.1) partitions the input space into finite number of equivalent partitions such that:

$$X = \bigcup_i P_i$$

with either  $P_i \subset A$  or  $P_i \subset X \setminus A$ , but can not be both. We also note that if  $P_i \subset A$  then  $P_{i+1} \subset X \setminus A$ . Clearly then  $A$  is a disjoint union of sets:

$$A = \bigcup a_i ; a_i \cap a_j = \emptyset ; i \neq j$$

### 3. PROBABILITY, CHAOS AND LEARNING ITERATIONS

We demonstrate that the general iterative learning system will be ‘almost surely’ [6][7] chaotic [8] [11].

**3.1. Properties of the The Learning Iteration.** We need some definitions to formally present the ideas discussed before.

**Definition 3.1. Rational Mapping of a Computable Function.**

Given an universal turing machine  $\mathbb{U}$  (definition A.13) any computable function can be encoded using the symbols in the tape of the machine. The rationalisation (definition A.16) of the tape  $\rho(T_C)$  then, serves as the rationalisation of the computable function.

$$\rho(C) = \rho(T_C, \mathbb{U}) = c ; C \in \mathbb{C}$$

where  $\mathbb{C}$  is the set of computable functions. This makes  $c \in [0, 1] \cap \mathbb{Q}$ .

**Definition 3.2. The Computable Functional.**

A computable function  $\mathcal{L} : \mathbb{Q} \rightarrow \mathbb{Q}$  is called a functional iff the range of the function can be interpreted as an encoding of a computable function. That is :

$$\forall x ; \exists C \in \mathbb{C} ; \rho(C) = \mathcal{L}(x)$$

**Definition 3.3. The Learning Iteration.**

Given a computable functional  $\mathcal{L} : \mathbb{Q} \rightarrow \mathbb{Q}$ , and input  $T_S$  (training set) the learning iteration is defined as :

$$c_{n+1} = \mathcal{L}(c_n)$$

with  $c_0 = \rho(T_S)$ . Here  $c_n$  signifies the rationalisation (definition 3.1) of the  $C_n \in \mathbb{C}$ .

**Theorem 3.1. Learning Iteration is Almost Surely Chaotic.**

Learning iteration, as defined as definition (3.3) is almost surely chaotic.

*Proof.* We note :  $\mathcal{L} : \mathbb{Q} \rightarrow \mathbb{Q}$ , from (theorem A.1) the theorem on chaos on computation, that almost every rational sequence is chaotic, the result is immediate.  $\square$

**Theorem 3.2. Almost Sure Non-Computability for converging function.**

The iteration of definition (3.3), when converges, almost surely converges to an un-computable function.

*Proof.* This is trivial from Real analysis. We know that the sequence  $c_n \in \mathbb{Q}$ , so to complete the space (definition A.4 ) we need to have the embedding space  $\mathbb{R}$ . Clearly then, almost all limit points would be irrational ( actually transcendental ), with a cardinality equal to continuum  $|\{ \lim < c_n > \}| = \aleph_1$  which is a well known theorem from real analysis [2]. Irrational numbers take infinitely many symbols to encode, therefore, the function limit  $\lim < c_n >$  can not be encoded in a Turing machine tape, and hence, is obviously non computable.

This is precisely what we wanted to show.  $\square$

**3.2. Properties of Self Similarity in Decision Tree.** We start with a definition of *self similarity*.

**Definition 3.4. Self Similarity.**

Let there be a topological space  $X$  (definition A.6) and a set of non-surjective homeomorphic functions  $\{f_s\}$  (definition A.8) indexed by a finite index set  $S = \{s\}$

with :

$$X = \bigcup_{s \in S} f_s(X) .$$

If  $X \subset Y$ , we call  $X$  self-similar if it is the only non-empty subset of  $Y$  such that the equation above holds for. Then  $\mathcal{S} = (X, S, \{f_s\})$  is called a self similar structure.

**Theorem 3.3. Infinite Binary Decision Trees have Self Similar Structure.**

A binary decision tree as defined in (2.1) is called infinite if it has countably infinite nodes. Accept set of such a tree exhibits self similar structure ( definition 3.4 ).

*Proof.* We note that every node in the tree first does a transform of the input space  $X$  using  $D_f$  into  $Y$  ; both homeomorphic to  $\mathbb{Q}$ . After that, this space  $Y$  is partitioned using  $D_d$  which are finite in number as discussed in section (2). Then this individual partitions are homeomorphic to  $\mathbb{Q}$ . Suppose  $n(D_d)$  defines the number of partitions. We can then say there are  $n(D_d)$  numbers of homeomorphic (on  $\mathbb{Q}$ ) non-surjective function available for each  $(D_d, D_f)$ . This is due to lemma (A.1).

The set of such functions is finite because  $\forall n ; n(D_d)$  and  $\mathbb{D}$  is finite. Suppose then, the set of such composition is termed as  $\mathbb{F}$ . It is then trivial that when  $X = \mathbb{Q}$  (at the root) then :

$$\mathbb{Q} = \bigcup_{F \in \mathbb{F}} F(\mathbb{Q})$$

Therefore, infinite binary decision tree have self similar structure.  $\square$

In fact it is well known that *Cantor Set* [9][10][8] is a generalisation of this sort of structure ( a dyadic Tree ).

**Theorem 3.4. Convergent ML Function would have Self Similar Structure.**

If an ML iteration (definition 3.3) converging to a function  $f_A$ , then accept set of  $f_A : A$  almost surely would have a self similar structure.

*Proof.* Almost surely the function is un-computable (theorem 3.2). We note that the due to the fixed size of the learner algorithm, the number of deciders the learning system can stays finite. And therefore, to be convergent the structure is to be extended to infinity : that is an infinite binary decision tree (definition 2.1). Now, using theorem (3.3), the result is immediate.  $\square$

#### 4. CHAOS AND MACHINE LEARNING - A SUMMARY

The theorems proven in the sections before can be useful to deduce very interesting properties of machine learning, which clearly showcases the problems arising from the chaotic nature of the Universal Computation.

- (1) The first phenomenon - that non converging ML process experienced by the co-author (section 1) is clearly explained by the inherent chaotic (theorem 3.1) nature of the ML iteration. The functions, encoded in the rational space, almost surely has a chaotic orbit, and that is why output of function  $C_n$  can drastically differ from  $C_{n+1}$ , given same input. However, we very well know that although almost all numbers are normal and transcendental proving them so is a problem, such is the problem here. Proving that the specific behaviour is chaotic can be done only in case to case basis.
- (2) For the second phenomenon reported [12] : As stated clearly in [12] that same ML algorithm from a different subset of the original training set facing the same problem. This is happening because in this case ML iteration is actually converging, and generating a fractal partition of the input space (theorem 3.4). This is precisely what they found : *input to output mapping mostly discontinuous* . They also noted that for each input which gets accepted, there are dense (definition A.7) set of inputs which gets rejected. This is a standard property for fractal space [8].

So, to summarise when the iteration converges, the result would ‘almost surely’ be a fractal set. Iteration from different initial conditions would, in fact converge to different indicator function, hence would accept different fractal sets  $A_1, A_2$  . Albeit  $A_1 \cap A_2 \neq \emptyset$  but also  $A_1 \Delta A_2 \neq \emptyset$  in fact the set  $A_1 \Delta A_2 \neq \emptyset$  should be dense (definition A.7) in  $X$ .

This clearly demonstrates the chaotic nature of the ML iteration convergence.

Finally we end with the same note they have : *indeed, any form of computable machinery would exhibit behaviour of this kind. These chaotic behaviours are in effect Universal*, as clearly demonstrated in this paper.

#### APPENDIX A. DEFINITIONS USED

##### **Definition A.1. Fixed Point of a function.**

For a function  $f : X \rightarrow X$  ,  $x^*$  is said to be a fixed point, iff  $f(x^*) = x^*$  .

##### **Definition A.2. Metric Space.**

A metric space is an ordered pair  $(M, d)$  where  $M$  is a set and  $d$  is a metric on  $M$  , i.e., a function:-

$$d : M \times M \rightarrow \mathbb{R}$$

such that for  $x, y, z \in M$  , the following holds:-

- (1)  $d(x, y) \geq 0$
- (2)  $d(x, y) = 0$  iff  $x = y$  .
- (3)  $d(x, y) = d(y, x)$
- (4)  $d(x, z) \leq d(x, y) + d(y, z)$

The function ‘ $d$ ’ is also called “distance function” or simply “distance”.

**Definition A.3. Cauchy Sequence in a Metric Space**  $(M, d)$  .

Given a Metric space  $(M, d)$  , the sequence  $x_1, x_2, x_3, \dots$  of real numbers is called ‘Cauchy Sequence’, if for every positive real number  $\epsilon$  , there is a positive integer  $N$  such that for all natural numbers  $m, n > N$  the following holds:-

$$d(x_m, x_n) < \epsilon.$$

Roughly speaking, the terms of the sequence are getting closer and closer together in a way that suggests that the sequence ought to have a limit  $x^* \in M$  . Nonetheless, such a limit does not always exist within  $M$  .

Note that by the term: *sequence* we are implicitly assuming *infinite sequence* , unless otherwise specified.

**Definition A.4. Complete Metric Space.**

A metric space  $(M, d)$  is called *complete* (or *Cauchy*) iff every Cauchy sequence (definition A.3) of points in  $(M, d)$  has a limit , that is also in  $M$  .

As an example of not-complete metric space take  $\mathbb{Q}$  , the set of rational numbers. Consider for instance the sequence defined by  $x_1 = 1$  and function  $d$  is defined by standard difference between  $d(x, y) = |x - y|$  , then :-

$$x_{n+1} = \frac{1}{2} \left( x_n + \frac{2}{x_n} \right)$$

This is a Cauchy sequence of rational numbers, but it does not converge towards any rational limit, but to  $\sqrt{2}$  , but then  $\sqrt{2} \notin \mathbb{Q}$  .

The closed interval  $[0, 1]$  is a Complete Metric space which is homomorphic (definition A.8) to  $\mathbb{R}$ .

**Definition A.5. [3] [4] Orbit.**

Let  $f : X \rightarrow X$  be a function. The sequence  $\mathcal{O} = \{x_0, x_1, x_2, x_3, \dots\}$  where

$$x_{n+1} = f(x_n) ; x_n \in X ; n \geq 0$$

is called an orbit (more precisely ‘forward orbit’) of  $f$ .

$f$  is said to have a ‘closed’ or ‘periodic’ orbit  $\mathcal{O}$  if  $|\mathcal{O}| \neq \infty$  .

**Definition A.6. Topological Space.**

Let the empty set be written as :  $\emptyset$ . Let  $2^X$  denotes the power set, i.e. the set of all subsets of  $X$ . A topological space is a set  $X$  together with  $\tau \subseteq 2^X$  satisfying the following axioms:-

- (1)  $\emptyset \in \tau$  and  $X \in \tau$  ,
- (2)  $\tau$  is closed under arbitrary union,
- (3)  $\tau$  is closed under finite intersection.

The set  $\tau$  is called a topology of  $X$ .

**Definition A.7. Dense Set.**

Let  $A$  be a subset of a topological space  $X$ .  $A$  is dense in  $X$  for any point  $x \in X$ , if any neighborhood of  $x$  contains at least one point from  $A$ .

The real numbers  $\mathbb{R}$  with the usual topology have the rational numbers  $\mathbb{Q}$  as a countable dense subset.

**Definition A.8. Homeomorphism.**

A function  $f : X \rightarrow Y$  between two topological spaces  $(X, T_X)$  and  $(Y, T_Y)$  is called a homeomorphism if it has the following properties:

- (1)  $f$  is a bijection (one-to-one and onto),
- (2)  $f$  is continuous,
- (3) the inverse function  $f^{-1}$  is continuous ( $f$  is an open mapping).

**Lemma A.1. Existence of Homeomorphic functions on Partitions.**

Let  $\mathbb{P} = \{P_i\}$  be a partition of  $X$  homeomorphic to  $\mathbb{Q}$ , such that:

$$\bigcup_i^n P_i = X ; \forall i \neq j \ P_i \cap P_j = \emptyset$$

then, there exists  $n = |\mathbb{P}|$  homeomorphic functions from  $X \rightarrow P_i$ .

**Definition A.9. Bounded Sequence.**

A sequence  $\langle x_n \rangle$  is called a bounded sequence iff :-

$$\forall n \ l \leq x_n \leq u ; -\infty < l \leq u < \infty$$

The number ‘ $l$ ’ is called the lower bound of the sequence and ‘ $u$ ’ is called the upper bound of the sequence.

**Lemma A.2. Bolzano-Weierstrass.**

Every bounded sequence has a convergent (Cauchy) subsequence.

It is to be noted that a bounded sequence may have many convergent subsequences (for example, a sequence consisting of a counting of the rationals has subsequences converging to every real number) or rather few (for example a convergent sequence has all its subsequences having the same limit).

**Definition A.10. Turing Machine.**

A “Turing Machine” is a 7-tuple  $(Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r)$ , where:-

- (1)  $Q$  is the set of states.
- (2)  $\Sigma$  is the set of input alphabets not containing the blank symbol  $\beta$ .
- (3)  $\Gamma$  is the tape alphabet, where  $\beta \in \Gamma$  and  $\Sigma \subseteq \Gamma$ .
- (4)  $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$  is the transition function.
- (5)  $q_0 \in Q$  is start state.
- (6)  $q_a \in Q$  is the accept state.
- (7)  $q_r \in Q$  is the reject state.

According to standard notion  $q_a \neq q_r$ , but we omit this requirement here, as we are not going to distinguish between two different types of halting (‘accept and halt’ vs ‘reject and halt’) of Turing Machines.

A Turing Machine ‘ $M$ ’ (definition A.10) computes as follows.



Initially ‘ $M$ ’ receives the input  $w = w_1w_2w_3...w_n \in \Sigma^*$  on the leftmost ‘ $n$ ’ squares on the tape, and the rest of the tape is filled up with blank symbol ‘ $\beta$ ’. The head starts on the leftmost square on the tape. As the input alphabet ‘ $\Sigma$ ’ does not contain the blank symbol ‘ $\beta$ ’, the first ‘ $\beta$ ’ marks end of input.

Once ‘ $M$ ’ starts, the computation proceeds wording to the rules of ‘ $\delta$ ’. However, if the head is already at the leftmost position, then, even if the ‘ $\delta$ ’ rule says move ‘ $L$ ’, the head stays there.

The computation continues until the current state of the Turing Machine is either  $q_a$  , or  $q_r$  . In lieu of that, the machine will continue running forever.

**Definition A.11. Decider Turing Machine.**

*A Turing Machine, which is guaranteed to halt on any input (i.e. reach one of the states  $\{q_a, q_r\}$  ) is called a decider.*

**Definition A.12. Undecidable Problem.**

*If for a given problem, it is impossible to construct a decider (definition A.11) Turing Machine, then the problem is called undecidable problem.*

**Definition A.13. Universal Turing Machine.**

*An ‘UTM’ or ‘Universal Turing Machine’ is a Turing Machine (definition A.10) such that it can simulate an arbitrary Turing machine on arbitrary input.*

**Definition A.14. Church Turing Thesis.**

*Every effective computation can be carried out by a Turing machine (definition A.10), and hence by an Universal Turing Machine(definition A.13).*

**Definition A.15. Gödelization (Gödel).**

*Any string from an alphabet set  $\Gamma$  can be represented as an integer in base ‘ $b$ ’ with  $b = |\Gamma|$ . To achieve this, create a one-one and onto Gödel map  $g : \Sigma \rightarrow D_b$  , where,*

$$D_b = \{0, 1, 2, \dots, b - 1\}.$$

*Gödelization or  $\mathbb{G} : \Sigma^+ \rightarrow \mathbb{Z}_+$  then, is defined as follows:*

*A string of the form  $w = w_{n-1}w_{n-2}...w_1w_0$  , with  $w_i \in \Gamma$  , can be mapped to an integer  $I_w = \mathbb{G}(w)$  as follows [11]:*

$$I_w = \mathbb{G}(w) = \sum_{k=0}^{n-1} g(w_k)b^k$$

The common decimal system is a typical example of Gödelization of symbols from  $\{0, 1, \dots, 8, 9\}$ . The binary system represents Gödelization of symbols from  $\{0, 1\}$ . As a far fetched example, any string from the whole english alphabet, can be written as a base 26 integers!

**Definition A.16. Rationalization.**

*Any string ‘ $w$ ’ of length ‘ $n$ ’ ( $|w| = n$ ) , created from an alphabet set  $\Gamma$ , can be represented as a rational number  $x \in \mathbb{Q}$ . We define the rationalization,  $\rho$  , in*

terms of Gödelization (definition A.15) as follows [11]:

$$x = \rho(w) = \frac{\mathbb{G}(w)}{b^n} = \mathbb{G}(w)b^{-n} = 0.w_{n-1}w_{n-2}\dots w_0$$

By definition,  $x \in [0, 1] \cap \mathbb{Q}$ .

**Theorem A.1. Bounded non repeating sequences are chaotic.**

Suppose  $\langle x_n \rangle$  is an infinite sequence such that  $x_n \in \mathbb{Q}$ , and  $x_i \neq x_j$  when  $i \neq j$ , then  $\langle x_n \rangle$  is chaotic. Given a bounded sequence, it is almost surely chaotic [11].

**Theorem A.2. Universal Computation is ‘Almost Surely’ chaotic.**

The rationalization of the sequences  $\langle T_n \rangle$  that is  $\langle \rho(T_n) \rangle$  of a Universal Turing machine is a bounded sequence between  $[0, 1] \cap \mathbb{Q}$  and hence ‘almost surely’ chaotic (theorem A.1) [11].

## REFERENCES

- [1] VAPNIK, V. , *The Nature of Statistical Learning Theory (Information Science and Statistics)* . Springer, Softcover reprint of hardcover 2nd ed. 2000 edition (October 21, 2010).
- [2] ROYDEN, H. L. , *Real Analysis* . Prentice Hall of India Private Limited, 2003.
- [3] ARROWSMITH, D.K. AND PLACE, C.M., *An Introduction to Dynamical Systems*. Cambridge University Press.
- [4] BRIN, MICHAEL AND STUCK, GARRETT., *Introduction to Dynamical Systems*. Cambridge University Press.
- [5] TURING, A. M., *On Computable Numbers, with an Application to the Entscheidungsproblem* . Proceedings of the London Mathematical Society, 2(42): 230-265
- [6] FELLER, W. , *An Introduction to Probability Theory and Its Applications*. Vol. 2, 3rd ed. New York: Wiley, 1968.
- [7] WILLIAMS, DAVID. , *Probability with Martingales*. Cambridge University Press, 1991.
- [8] HEINZ-OTTO PEITGEN, HARTMUT JÜRGENS , DIETMAR SAUPE, *Chaos and Fractals New Frontiers of Science*. Springer.
- [9] SMITH, HENRY J.S. *On the integration of discontinuous functions*. Proceedings of the London Mathematical Society, Series 1, vol. 6, pages 140-153, 1874.
- [10] CANTOR, GEORGE. *Über unendliche, lineare Punktmannigfaltigkeiten V [On infinite, linear point-manifolds (sets)]*. Mathematische Annalen, vol. 21, pages 545-591, 1883.
- [11] MONDAL , NABARUN AND GHOSH, PARTHA P , *Universal Computation is ‘Almost Surely’ Chaotic* . Theoretical Computer Science, 2014 , doi:10.1016/j.tcs.2014.07.005
- [12] CHRISTIAN SZEGEDY, WOJCIECH ZAREMBA, ILYA SUTSKEVER, JOAN BRUNA, DUMITRU ERHAN, IAN GOODFELLOW, ROB FERGUS, *Intriguing properties of neural networks* . Arxiv <http://arxiv.org/abs/1312.6199>

D.E.SHAU & CO. INDIA, HYDERABAD  
E-mail address: mondal@deshaw.com

MICROSOFT INDIA, HYDERABAD  
E-mail address: parthag@microsoft.com